# Learning Social Navigation from Internet Videos in the Policy State Space

Jiaming Wang[1,*], Duc Thang Nguyen[1], Jizhuo Chen[1], Volodymyr Shcherbyna[2], Diwen Liu[3], Zhengcheng Shen[4], and Harold Soh[1]

***Abstract*— Training robust social-navigation policies requires simulators with diverse scene layouts, terrain, and human motion, but constructing such environments and specifying pedestrian behavior is costly. We propose an efficient pipeline that converts ordinary monocular walking videos directly into closed-loop social-navigation training environments in the policy's state space. Our key observation is that local social navigation primarily depends on two types of information: where the robot can traverse and how nearby pedestrians move. We therefore represent the static scene as a metric traversability map, which can be rigidly transformed under counterfactual robot motion, while directly replaying the pedestrian trajectories recovered from the video over time. This abstraction allows us to define the forward dynamics directly in the policy's state space and efficiently simulate counterfactual robot states without reconstructing or rendering photorealistic observations. The resulting policy achieves 81.2% success in the independent Arena benchmark, compared with 75.0% for the strongest baseline, and succeeds in 19/20 real-robot trials without policy fine-tuning.**

## I. Introduction

Social-navigation policies are commonly trained in simulation because large-scale interaction on real robots is costly and potentially unsafe. These simulators typically use manually constructed environments and pedestrians driven by behavior models such as social force or learned trajectory predictors [1], [2]. The diversity of environments is therefore constrained by the cost of constructing layouts, terrain, and obstacles, and the diversity of pedestrian behavior by the chosen human model. Video-to-simulation and generative scene methods increase environment diversity by reconstructing or generating realistic scenes from visual data [3]–[6], but they require a renderable 3D scene representation, such as reconstructed geometry, Gaussian splats, or generated assets, and render counterfactual observations during simulation, which can be computationally expensive. For social navigation, dynamic pedestrians further require modeling their appearance and motion.

In this work, we build closed-loop social-navigation training environments directly from ordinary monocular walking videos. A social-navigation policy primarily needs to know where it can move and how nearby pedestrians move, so we represent each video directly in the policy's state space: a metric traversability map of the static scene together with the pedestrian trajectories recovered from the video (Fig. 1). During simulation, the map is rigidly transformed with the robot pose, while pedestrians follow their recorded trajectories. This enables reinforcement learning under counterfactual robot actions without reconstructing or rendering the visual scene.

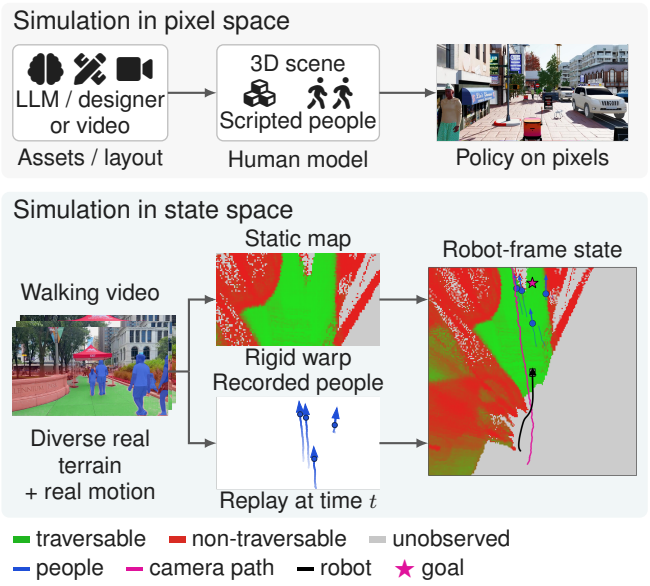



Fig. 1. **From video to social-navigation simulation.** Ordinary monocular walking videos become closed-loop training environments in the policy's state space. The recorded world is decomposed into a static traversability map, which is rigidly transformed under counterfactual robot motion, and dynamic pedestrian trajectories, which are replayed over time, so reinforcement learning requires no reconstructed or rendered photorealistic scene.

The formulation is lightweight and scalable. We construct 6,075 training episodes from 33.8 hours of walking videos spanning diverse cities, terrain, and crowds, and the simulator runs at roughly 1,000 environment steps/s on a single RTX 5090. Ablations show that directly replaying recorded pedestrian trajectories is sufficient to learn robust social navigation without an explicit human behavior model. A policy trained solely in these video-derived environments transfers without fine-tuning to the independent Arena simulator, where it reaches 81.2% success and outperforms prior methods, and to a physical robot, where it succeeds in 19/20 trials.

Our contributions are threefold: (1) a state-space simulator that converts ordinary walking videos into closed-loop social-navigation training environments while preserving the

[1]National University of Singapore, Singapore. [2]Singapore Management University, Singapore. [3]The Chinese University of Hong Kong, Hong Kong SAR, China. [4]Max Planck Institute for Plasma Physics, Germany.
*Corresponding author: jiaming@comp.nus.edu.sg

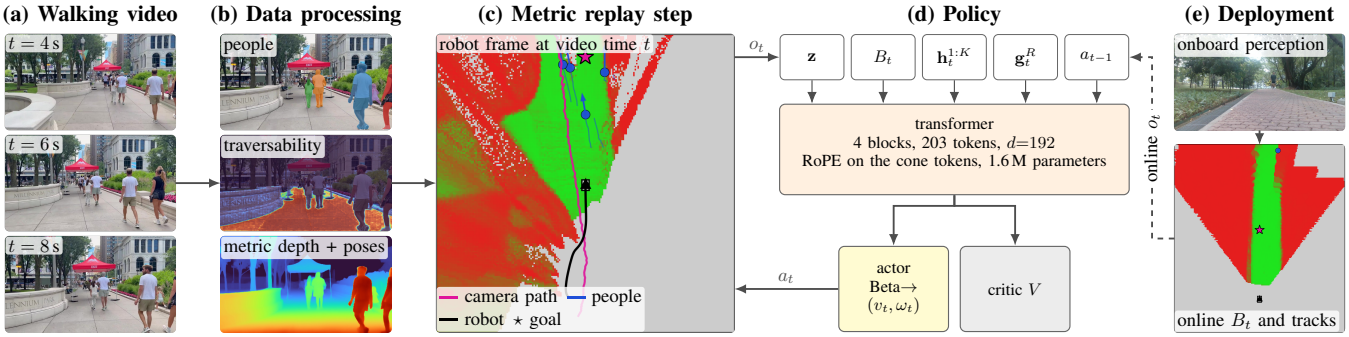



Fig. 2. **Overview.** (a) First-person walking videos are processed by (b) pretrained perception models to recover a gravity-aligned metric traversability map and time-indexed pedestrian states. (c) These quantities define the state-space replay simulator: world time follows the recording, while the robot pose evolves independently from the policy commands. The recorded map and pedestrians are transformed into the robot frame at each step, enabling counterfactual motion away from the camera path, collision checking, and closed-loop training without rendering. (d) The policy encodes the egocentric traversability map, nearby pedestrian states, goal, and previous action with a lightweight transformer and outputs velocity commands. (e) At deployment, the same policy state is reconstructed online from onboard perception, allowing the video-trained policy to transfer without modification.

recorded pedestrian motion; (2) a systematic study of policy learning in these environments, including training strategy, pedestrian modeling, and simulator design; and (3) evaluation on unseen videos, the Arena benchmark, and a physical robot, demonstrating transfer without policy fine-tuning. The processed dataset, trained policy checkpoints, and code will be released upon publication.

## II. Related Work

**Social navigation.** Social-navigation methods have evolved from reciprocal collision avoidance to learned policies that model pedestrian interactions through reinforcement learning, interaction graphs, and trajectory prediction [7]–[13]. Most are trained with synthetic pedestrian behavior, while real-world datasets [14]–[16] provide natural interactions but are costly to collect at scale. Rather than proposing another policy architecture, we focus on obtaining scalable closed-loop training from real pedestrian behavior in video.

**Simulation for social navigation.** Simulators such as HuNavSim and Arena [17], [18] combine constructed or scanned environments with reactive crowds driven by social force or learned trajectory predictors [1], [2], so the pedestrian behavior is bounded by the chosen model. SocNavBench [19] replays prerecorded pedestrian trajectories, but as an evaluation benchmark with a fixed set of prepared scenarios rather than diverse training environments. We instead construct both traversability and pedestrian motion from ordinary monocular videos, enabling closed-loop training across diverse real-world scenes.

**Video-to-simulation.** Video-to-simulation methods reconstruct or generate realistic environments using Gaussian splats, 3D assets, or generative models [3]–[6], [20]; dynamic social scenes additionally require human representations and motion models, making construction and rendering costly. Other works use web videos as navigation demonstrations [21]–[24], without closed-loop interaction beyond the recorded trajectories. We instead extract only the task-relevant state from video to build efficient closed-loop training environments.

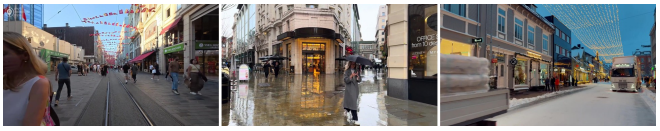


Fig. 3. Frames from three held-out source videos: an evening crowd on Istiklal Street (Istanbul), a rainy shopping street (London), and a snow-covered street at dusk (Tromsø). Ordinary first-person walking tours are the only input of the pipeline.

## III. Method

### A. Problem statement and overview

We consider a wheeled or legged robot with a forward RGB-D camera that must reach a fixed world-frame goal $\mathbf{g}^W \in \mathbb{R}^2$ while avoiding static obstacles and pedestrians. At 10 Hz, the policy $\pi_\theta$ maps an observation $o_t$ to a velocity command $a_t = (v_t, \omega_t)$. Rather than pixels, the observation contains an egocentric traversability map $B_t \in [0,1]^{200\times200}$ covering $\pm10$ m at 0.1 m resolution, up to $K = 20$ pedestrian states $\mathbf{h}_t^i = (x, y, v_x, v_y)$ within 10 m, the goal $\mathbf{g}_t^R$ in the robot frame, and the previous action.

This representation captures the quantities that determine local navigation outcomes: which ground is traversable, and where nearby people are and are moving. It also makes video-derived simulation simple: the static scene changes under robot motion only by a rigid transform, while pedestrian states can be replayed directly from the video (Fig. 2). Constructing it from monocular video requires all observations in a common metric frame and a *temporally consistent* world state: the map the robot sees at video time $t$ must not contain ground that the camera only observed later. Each 20 s, 10 fps clip (Fig. 3) is processed accordingly.

### B. Data processing pipeline

**Metric geometry and traversability.** Depth Anything 3 [25] is run once over the 200 frames of a clip and jointly estimates metric depth, intrinsics, and camera poses. Because navigation is governed by the ground plane rather than raw 3D geometry, SAM-TP [26] predicts pixel-wise walkability, whose high-confidence regions are back-projected with the estimated depth; a RANSAC ground

TABLE I
CORPUS STATISTICS.

| | Train | Test |
|---|---|---|
| Source videos (distinct cities) | 25 (12) | 19 (17) |
| 20 s episodes extracted | 8,147 | 3,126 |
| Episodes passing gates | 6,075 | 2,155 |
| Hours of usable video | 33.8 | 12.0 |
| Pedestrian tracks | 91,261 | 29,846 |
| Pedestrians per frame (mean / max) | 2.5 / 18 | 2.3 / 18 |
| Demonstrator path per episode (m) | 23.7 | 21.7 |

plane fitted in the best-grounded frames defines the gravity direction and a metric camera height (median 1.3 m). The resulting gravity-aligned world frame $W$ expresses navigation, and the projected camera centers and optical axes give the demonstrator poses $(x_t, y_t, \psi_t)$.

**Causal traversability map.** For each frame, depth points outside person boxes are transformed into $W$ and rasterized at 0.1 m with their traversability scores. Measurements are accumulated only from frames up to the current time with an exponential moving average, producing a map sequence $\{M_t\}$ in which $M_t$ contains only cells observed by frame $t$. Only map accumulation is causal: depth, poses, and the ground plane are estimated from the full clip and are thus more accurate than those available to an online robot. Unobserved cells receive the free prior (traversability 1) in both the observation and the collision map, making unobserved space indistinguishable from observed free space; we examine this assumption in Sec. III-D.

**Pedestrian trajectories.** YOLO11 [27] segmentation and pose estimates detect visible pedestrians, and BoT-SORT [28] associates them over time. Each person is localized on the ground plane by back-projecting its mask with metric depth, and a constant-velocity Kalman filter per identity smooths the track and estimates velocity; a track is created after four consecutive in-range detections with a visible torso and coasts through at most eight missed frames. The time-indexed states $(x, y, v_x, v_y)_t$ provide the pedestrian dynamics of the simulator.

**Episode gating and cost.** We retain clips with sufficient camera motion, valid map reconstruction, and meaningful pedestrian interaction, removing stationary segments, pose failures, and scenes without nearby people; 73% of candidate clips pass (Table I). Every stage is feed-forward, and converting one 20 s clip takes 4.5 min on one A100 GPU.

### C. The state-space simulator

Each episode contains the causal map sequence $\{M_t\}$, pedestrian states $\{P_t\}$ in $W$, and the demonstrator trajectory. The simulator decouples *world time* from *robot motion*: the recorded scene advances along the video timeline, while the robot pose $\mathbf{p}_t = (x, y, \psi)$ evolves according to the policy. The two interact only when the recorded state is transformed into the robot's current frame, so the robot can take actions never observed in the video without predicting or rendering new images.

**World and robot evolution.** At step $t$, the simulator retrieves $M_{f_0+t}$ and $P_{f_0+t}$, where $f_0$ is the start frame, preserving the real scene geometry and the pedestrians' recorded trajectories, speeds, and timing. The robot pose is integrated from $a_t = (v_t, \omega_t)$ with a constant-twist unicycle model over $\Delta t = 0.1$ s:

$$\psi' = \psi + \omega \Delta t,$$
$$x' = x + \tfrac{v}{\omega}(\sin\psi' - \sin\psi), y' = y + \tfrac{v}{\omega}(\cos\psi - \cos\psi'), \quad (1)$$

with the straight-line limit as $|\omega| \to 0$. A static collision is declared when the 20th percentile of the map values under the $0.7 \times 0.5$ m footprint falls below 0.1; pedestrians are 0.25 m discs. Inverting the same dynamics between successive demonstrator poses gives the imitation targets used in Sec. IV-B.

**Observation synthesis.** The observation samples $M_{f_0+t}$ on a $200 \times 200$ grid centered and oriented at $\mathbf{p}_t$, and pedestrian positions and velocities are transformed into the same frame, keeping the $K$ nearest tracks within 10 m. Because this is defined for any $\mathbf{p}_t$, the robot can stop, turn, or leave the demonstrator path while observing the corresponding recorded world state.

**Start and goal.** During training, the start is jittered within a 2 m forward half-disc around the demonstrator pose, and the goal is the first demonstrator position at least 15 m away, jittered within 2 m; both must be free. These perturbations prevent the policy from simply replaying the camera trajectory while keeping the episode close to the demonstration. Evaluation uses the unjittered start and goal.

### D. Simulation assumptions

Counterfactual robot motion introduces two approximations that a single recording cannot resolve.

**Partial static-scene coverage.** The robot can enter regions the video never observed, which carry the free prior. Starts and goals near the recorded route, together with the progress reward and step penalty, discourage such detours. On held-out episodes (Table II), the robot's local maps are as well supported by measurements as those of the recorded walker (21% vs. 23% observed cells), and the fraction of path traveled over never-observed cells rises only moderately (15.2% vs. 11.2%). An unknown-aware variant that marks unobserved cells in the input and penalizes footprint overlap with them lowers this fraction from 14.4% to 12.7% but also lowers success from 91.4% to 87.4%, suggesting that the policy does not substantially rely on unsupported space and that penalizing all unobserved regions restricts navigation unnecessarily.

**Non-reactive pedestrian motion.** Pedestrians follow their recorded trajectories rather than reacting to the robot. Replay nevertheless preserves the real speeds, crossings, and crowd configurations without an additional behavior model. We test this approximation in Sec. IV-B by training with reactive social-force and Trajectron++ pedestrians and evaluating under all three dynamics. Replay training remains robust to reactive pedestrians and performs better overall than either model-based alternative, and the same holds in Arena (Sec. IV-C) and on the real robot (Sec. IV-D), suggesting that preserving diverse recorded human motion is more beneficial

TABLE II

OBSERVATION COVERAGE ON HELD-OUT EPISODES: FRACTION OF LOCAL-MAP CELLS SUPPORTED BY RECORDED MEASUREMENTS, DISTANCE TRAVELED ON UNOBSERVED SPACE (M/EPISODE AND % OF PATH LENGTH), AND SUCCESS RATE.

| Policy | Observed | Unsupported path | | Success |
|---|---|---|---|---|
| | cell (%) | (m/ep.) | (%) | (%)↑ |
| Recorded walker | 23 | 1.7 | 11.2 | – |
| Ours (released) | 21 | 2.3 | 15.2 | 92.6 |
| Control (30 M) | 21 | 2.2 | 14.4 | 91.4 |
| Unknown-aware (30 M) | 21 | 1.8 | 12.7 | 87.4 |

than approximating it with the pedestrian models considered here.

### E. Policy

The policy operates directly on the state representation (Fig. 2d). We divide the 10 m neighborhood into 180 angular cones of 2° and encode the near-to-far traversability values of each cone (at most 183 cells, padded) as one token through a linear layer. These map tokens are combined with up to $K = 20$ pedestrian tokens, a goal token, a previous-action token, and a readout token $\mathbf{z}$, and processed by a lightweight 4-block transformer (203 tokens of width 192; each block applies RMSNorm, attention with 6 query heads and one shared key/value head, and a SiLU feed-forward layer of width 768; 1.6 M parameters). Rotary position encoding [29] is applied only to the ordered map tokens, so pedestrian tokens remain permutation-invariant. The readout token feeds a critic and an actor that parameterizes independent Beta distributions [30], affinely scaled to $v \in [-1.3, 1.3]$ m/s and $\omega \in [-2, 2]$ rad/s, which avoids action clipping during PPO.

### F. Training with reinforcement learning

We train the policy from scratch with PPO [31] in the state-space simulator, whose counterfactual starts and closed-loop dynamics expose it to states and actions beyond the recorded camera trajectory. We also tried initializing the policy by imitating the recovered camera motion, but controlled ablations show no benefit and slightly worse final performance (Sec. IV-B). Reaching within 0.3 m of the goal gives $+5$, and a collision with a pedestrian or non-traversable ground terminates the episode with $-5$. At all other steps,

$$r_t = 0.5(d_{t-1} - d_t) - 0.01 - 0.05\max(0, -v_t) - 0.005\max(0, |\omega_t - \omega_{t-1}| - 0.7), \quad (2)$$

where $d_t$ is the planar distance to the goal. The terms reward progress, penalize slack and backward motion, and discourage abrupt yaw-rate changes beyond a 0.7 rad/s deadband. Episodes end on success or collision and truncate when the video ends or after 180 steps. We use 32 parallel environments, clipping 0.2, $\gamma = 0.99$, GAE $\lambda = 0.95$, and a learning rate annealed from $5 \times 10^{-5}$ to $10^{-5}$; a 100 M-step run takes 26 h on one RTX 5090.

TABLE III

CLOSED-LOOP EVALUATION ON THE HELD-OUT EPISODES IN THE STATE-SPACE SIMULATOR.

| | |
|---|---|
| Success rate (95% Wilson interval) | **92.6%** [91.4, 93.7] |
| Pedestrian / static-obstacle collision | 3.7% / 1.9% |
| Timeout (18 s) | 1.8% |
| SPL / path efficiency (shortest / traveled) | 0.860 / 0.928 |
| Time to goal (s) / mean speed (m/s) | 12.2 / 1.28 |
| Time in personal zone ($< 1.2$ m) per episode (s) | 0.24 |

## IV. EXPERIMENTS

We evaluate whether state-space simulation produces a policy that (i) generalizes to unseen walking videos, (ii) benefits from the proposed simulator and training design, (iii) transfers to an independent simulator and competes with conventionally trained social-navigation policies, and (iv) transfers to a physical robot without fine-tuning.

### A. Data and held-out evaluation

We collect 44 public 4K first-person walking-tour videos spanning 29 cities (Fig. 3). Training and test videos are disjoint, with no city shared between the two splits (Table I), yielding 6,075 training and 2,155 test episodes with roughly 121k pedestrian tracks, far more natural social encounters than robot-collected datasets [14], [15], while retaining the layouts, terrain, and crowd densities of real streets.

We first evaluate closed-loop on every held-out episode in the state-space simulator: the robot starts at the demonstrator's pose at $t = 2$ s and must reach the first demonstrator position at least 15 m ahead among the recorded people. The policy succeeds in 92.6% of episodes from 19 unseen videos (Table III), with paths within 8% of the straight-line length (SPL [32], [33] 0.860), and spends a quarter of a second per episode inside anyone's personal space. Half the remaining failures are pedestrian collisions. Failures concentrate in a few scenes with dense dynamic traffic, such as a Zurich shopping street with trams and cyclists (27 of 159 failures) and evening walks in Paris (19), Barcelona, and Istanbul. For reference, the recorded camera trajectory itself satisfies the simulator's collision criteria in only 72% of episodes, with 21% pedestrian and 7% static collisions.

### B. Ablations on held-out videos

We study the main training and simulator choices on the 2,155 held-out episodes, training each variant for 30 M steps (Table IV). Besides the PPO-from-scratch control, we evaluate imitation learning (IL) of the recovered demonstrator actions and PPO initialized from IL. We also remove spawn jitter, restrict the map input to the current 90° forward field of view, add a graded proxemics penalty, replace the angular-cone tokens with $2 \times 2$ m square patches, vary the amount of training video, and replace replayed pedestrians with reactive social-force (SF) or Trajectron++ dynamics.

**Replay is an effective approximation of pedestrian behavior.** We compare replay against two reactive alternatives. For SF [1], each pedestrian's preferred motion follows its recorded trajectory, so pedestrians reproduce the observation

TABLE IV

ABLATIONS ON HELD-OUT EPISODES. SUCC. IS SUCCESS RATE, PED. COLL. AND STAT. COLL. ARE PEDESTRIAN AND STATIC-OBSTACLE COLLISION RATES, AND PERS. IS TIME PER EPISODE WITHIN 1.2 M OF A PEDESTRIAN. BRACKETS GIVE 95% WILSON CONFIDENCE INTERVALS. ALL VARIANTS ARE TRAINED FOR 30 M STEPS.

| Variant | Succ.↑ [95% CI] | SPL↑ | Ped. coll.↓ | Stat. coll.↓ | Pers. (s)↓ |
|---|---|---|---|---|---|
| Control | 91.4 [90.1, 92.5] | 0.858 | 4.4 | 1.5 | 0.19 |
| IL only | 55.3 [53.2, 57.4] | 0.548 | 32.1 | 11.2 | 0.46 |
| PPO from IL initialization | 90.0 [88.6, 91.2] | 0.859 | 5.4 | 1.2 | 0.45 |
| No spawn jitter | 90.4 [89.1, 91.6] | 0.860 | 5.4 | 1.6 | 0.58 |
| 90° FOV map | 88.9 [87.5, 90.1] | 0.846 | 4.5 | 1.3 | 0.09 |
| Proxemics reward | 85.0 [83.4, 86.5] | 0.823 | 11.5 | 2.1 | 0.42 |
| Square-patch tokens | 89.7 [88.3, 90.9] | 0.855 | 5.4 | 1.9 | 0.39 |
| 25% training videos | 81.8 [80.1, 83.3] | 0.791 | 13.5 | 2.4 | 0.28 |
| 50% training videos | 88.2 [86.8, 89.5] | 0.834 | 5.0 | 0.9 | 0.24 |
| Social-force pedestrians | 75.6 [73.7, 77.4] | 0.738 | 21.1 | 1.6 | 0.44 |
| Trajectron++ pedestrians | 90.0 [88.7, 91.2] | 0.850 | 5.7 | 1.8 | 0.34 |
| *Same episodes, pedestrians driven by social-force model* | | | | | |
| Control | 93.4 [92.3, 94.4] | 0.872 | 1.8 | 1.5 | 0.18 |
| Social-force pedestrians | 88.7 [87.3, 89.9] | 0.859 | 7.6 | 1.4 | 0.59 |
| Trajectron++ pedestrians | 93.3 [92.1, 94.3] | 0.878 | 2.3 | 1.7 | 0.32 |
| *Same episodes, pedestrians driven by Trajectron++ model* | | | | | |
| Control | 91.9 [90.7, 93.0] | 0.866 | 4.1 | 1.5 | 0.20 |
| Social-force pedestrians | 75.5 [73.6, 77.2] | 0.738 | 21.8 | 1.3 | 0.46 |
| Trajectron++ pedestrians | 91.3 [90.0, 92.4] | 0.870 | 5.2 | 1.6 | 0.37 |

when unperturbed but react to the robot and one another. As a learned alternative, we train Trajectron++ [2] from scratch on the pedestrian tracks and camera carrier of the training videos (5,629 scenes, 82,880 tracks), since the released ETH/UCY model transfers poorly to walking-tour data. Agents are resampled at 0.4 s, and the model uses up to 2.8 s of history to predict 1.2 s of future motion; in simulation it is conditioned on the simulated histories of both pedestrians and robot. Across all three evaluation dynamics, the replay-trained policy is consistently robust, so training on recorded motion generalizes even when pedestrians become reactive at test time. Training with SF pedestrians performs substantially worse, while Trajectron++ is competitive with replay but offers no clear benefit despite an additional learned model and slower simulation (about 800 vs. 1,000 steps/s). Direct trajectory replay thus provides a simple, efficient, and sufficiently diverse training distribution without an explicit reactive pedestrian model.

**More video improves social navigation.** Using 25%, 50%, and 100% of the training videos gives 81.8%, 88.2%, and 91.4% success. The clearest change is in pedestrian collisions, which fall from 13.5% to 5.0% and then 4.4%, while static collisions show no comparable trend: more video mainly exposes the policy to more diverse crowd configurations and pedestrian behaviors.

**Closed-loop interaction is essential.** Imitating the recovered demonstrator motion reaches only 55.3% success, and initializing PPO from the imitation policy provides no benefit (90.0%). The value of video is thus not merely as demonstrations: an interactive environment exposes the policy to off-trajectory states, failures, and recoveries that are absent from imitation.

**Other design choices** have modest effects: the 90° field-of-view map and square patches reduce success to 88.9% and 89.7%. Removing spawn jitter changes success by only 1.0 point (90.4%), so the method does not depend strongly on this augmentation. A graded proxemics penalty ($-0.1$ within 0.45 m, decreasing linearly to 0 at 1.2 m) degrades success to 85.0% and more than doubles pedestrian collisions.

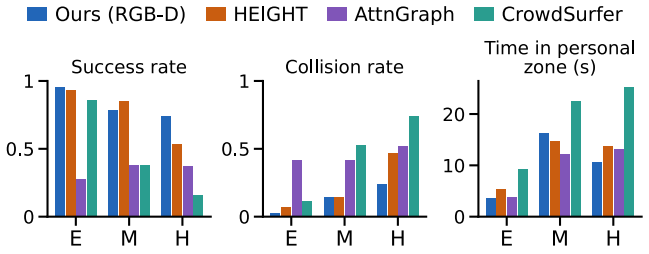



Fig. 4. Arena results by crowd density: E, M and H are the easy (0–4 pedestrians), medium (5–9) and hard (10–14) scenarios. The advantage of our video-trained policy grows with density.

### C. Arena benchmark

**Setup.** We test transfer beyond our simulator in Arena [18], with photorealistic Isaac Sim environments and reactive social-force pedestrians. Four UrbanVerse street scenes each contain 40 scenarios (ten frontal, five same-direction, five crossing, and twenty mixed encounters) with 0–4 (*easy*), 5–9 (*medium*), or 10–14 (*hard*) pedestrians. A Clearpath Jackal with an OAK-D RGB-D camera navigates between goals 18–22 m apart; success requires reaching within 1 m of the goal within 180 s, and any contact terminates the episode. We compare against HEIGHT [12], AttnGraph [11], and CrowdSurfer [13] with their official checkpoints and Arena integrations. All three follow subgoals from a Nav2 global path, and HEIGHT and CrowdSurfer also receive 360° LiDAR, whereas our policy builds its map from RGB-D and odometry without LiDAR, prebuilt map, or global planner. Since the simulator does not provide the LiDAR intensities required by the baselines' pedestrian detector, and to isolate navigation from detection quality, all methods receive ground-truth pedestrian positions with Kalman-filtered velocities. Each of our three variants is trained for 200 M steps, and the checkpoint with the highest rollout reward runs at 10 Hz without fine-tuning.

**Results.** Trained solely in video-derived replay environments, our policy achieves the highest success (81.2%) and SPL (0.747) (Table V). Compared with HEIGHT, the strongest baseline, it improves success by 6.2 points and reduces collisions from 25.0% to 15.0%, for both pedestrians (13.8% vs. 19.4%) and walls (1.2% vs. 5.6%). The advantage grows with crowd density: on hard scenarios, our policy reaches 74.2% success, compared with 53.2% for HEIGHT, 37.1% for AttnGraph, and 16.1% for CrowdSurfer (Fig. 4). Its time in the personal and intimate zones (10.6 s and 2.1 s per episode) is not the lowest among all methods but is below HEIGHT's 11.8 s and 3.5 s, suggesting that replay training teaches the policy to keep making progress through dense crowds, including the brief close but non-colliding

TABLE V

ARENA/ISAAC SIM RESULTS ON 160 SCENARIOS. PERSONAL AND INTIMATE DENOTE SECONDS PER EPISODE WITHIN 1.2 M AND 0.45 M OF A PEDESTRIAN, RESPECTIVELY.

| | Inputs | | Outcome (%) | | | Efficiency | | | Social | |
|---|---|---|---|---|---|---|---|---|---|---|
| Method | LiDAR | Plan | Success [95% CI]↑ | Coll.↓ (ped/wall) | Timeout↓ | Time (s)↓ | SPL↑ | $\bar{v}$ (m/s) | Personal (s)↓ | Intimate (s)↓ |
| CrowdSurfer [13] | ✓ | ✓ | 42.5 [35.1, 50.2] | 50.0 (47.5/2.5) | 7.5 | 46.2 | 0.418 | 0.49 | 19.9 | 1.9 |
| AttnGraph [11] | | ✓ | 35.0 [28.0, 42.7] | 45.6 (11.2/34.4) | 19.4 | 52.0 | 0.299 | 0.47 | 10.3 | **1.2** |
| HEIGHT [12] | ✓ | ✓ | 75.0 [67.8, 81.1] | 25.0 (19.4/5.6) | **0.0** | 46.4 | 0.714 | 0.47 | 11.8 | 3.5 |
| OURS (SF peds) | | | 58.1 [50.4, 65.5] | 41.9 (39.4/2.5) | **0.0** | **42.6** | 0.559 | 0.49 | **8.4** | 1.8 |
| OURS (Trajectron++ peds) | | | 70.0 [62.5, 76.6] | 25.0 (20.0/5.0) | 5.0 | 51.9 | 0.599 | 0.48 | 12.2 | 2.3 |
| OURS | | | **81.2** [74.5, 86.5] | **15.0** (13.8/1.2) | 3.8 | 46.2 | **0.747** | 0.49 | 10.6 | 2.1 |



Fig. 5. (a) Unitree Go2 with RGB-D camera and LiDAR; (b, c) the indoor and outdoor test scenes.

TABLE VI

REAL-ROBOT RESULTS (10 TRIALS PER METHOD AND SCENE; MEANS OVER SUCCESSES).

| Scene | Method | Success↑ | Mean time (s) | Mean path (m) |
|---|---|---|---|---|
| Indoor | Nav2-MPPI | 5/10 | 28.6 | 9.64 |
| | OURS (Traj. peds) | 7/10 | 32.1 | 10.96 |
| | OURS | 9/10 | 31.1 | 10.14 |
| Outdoor | Nav2-MPPI | 3/10 | 18.9 | 7.74 |
| | OURS (Traj. peds) | 8/10 | 16.1 | 8.29 |
| | OURS | 10/10 | 17.9 | 8.59 |

passes common in walking videos, without becoming overly conservative or relying on unsafe proximity.

Arena also tests the pedestrian-motion approximation out of distribution. The SF- and Trajectron++-trained variants reach only 58.1% and 70.0% success. The Trajectron++ result is particularly informative: although it is trained on the same extracted trajectories and nearly matches replay on held-out videos, its advantage does not carry over to Arena; a learned model necessarily approximates the empirical behavior distribution and may smooth rare interactions or accumulate error during rollout, whereas replay preserves the original speeds, crossings, and yielding behavior. Replay is thus not only simpler and faster but also transfers more robustly to unseen pedestrian dynamics.

### D. Real-robot deployment

**Setup.** We deploy the policies on a Unitree Go2 quadruped with an RGB-D camera and LiDAR (Fig. 5a). The traversability state is built online from RGB-D and odometry, and pedestrians are initialized when a YOLO11 detection and a DBSCAN cluster of unfiltered LiDAR points agree, then tracked from LiDAR with a Kalman filter and Hungarian association on Euclidean distance. Commands are scaled to the robot's speed limit, and no global map, planner, or fine-tuning is used. We compare against Nav2's MPPI controller [34], which uses the same traversability memory as a graded costmap with LiDAR obstacles overlaid, since a geometric LiDAR costmap cannot distinguish pavement from grass. MPPI treats detected pedestrians as occupied cells, whereas the learned policies receive tracked positions and velocities. All methods share the same velocity smoother and collision monitor. We also test the Trajectron++ variant.

**Protocol and results.** In one indoor and one outdoor scene (Fig. 5b, c), the robot navigates to a goal 8–10 m away through five scripted encounters, each repeated twice per method; a trial succeeds if the robot reaches the goal without static collision or coming within 0.3 m of a pedestrian. The replay-trained policy succeeds in 19/20 trials, compared with 15/20 for the Trajectron++ variant and 8/20 for MPPI (Table VI). MPPI often keeps its course until a moving pedestrian enters the local costmap, causing late avoidance, whereas the learned policies use velocity estimates to slow down or move aside before the crossing point. The replay-trained policy is the most reliable despite never seeing a reactive pedestrian model during training, consistent with the held-out-video and Arena results. The single replay failure occurs indoors, where the 0.5 m/s hardware speed limit, well below typical training speeds, leaves too little time to leave an approaching person's path.

## V. CONCLUSION, LIMITATIONS, AND FUTURE WORK

We presented a scalable approach to learning social navigation from monocular walking videos by constructing the simulator directly in the policy's state space. Reconstructed traversability and replayed pedestrian trajectories enable counterfactual actions and reinforcement learning *without* image rendering, and the resulting policy generalizes to unseen videos and transfers to an independent simulator, with higher success and fewer pedestrian collisions than prior methods, and to a real robot without modification.

Our representation assumes approximately planar ground within each $20 \times 20$ m local window; substantial curvature or elevation changes can distort the BEV projection. We also model motion only for pedestrians and treat the remaining scene as static, which suits pedestrian paths, but moving vehicles and bicycles can introduce map errors and contribute to failures in mixed-traffic scenes. Finally, pedestrian states depend on human-pose detection and depth, whose errors degrade velocity estimates and hence collision avoidance. These limitations are interesting avenues for future work.

## REFERENCES


[1] D. Helbing and P. Molnár, "Social force model for pedestrian dynamics," *Physical Review E*, vol. 51, no. 5, pp. 4282–4286, 1995.

[2] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in *Proc. European Conf. Computer Vision (ECCV)*, 2020, pp. 683–700.

[3] Z. Xie, Z. Liu, Z. Peng, W. Wu, and B. Zhou, "Vid2Sim: Realistic and interactive simulation from video for urban navigation," in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR)*, 2025.

[4] S. Yoo, Y. Jang, D. Kim, Y. Han, S. Jung, and H. J. Kim, "ReaDy-Go: Real-to-sim dynamic 3D Gaussian splatting simulation for environment-specific visual navigation with moving obstacles," *IEEE Robotics and Automation Letters*, 2026.

[5] M. Liu, H. He, E. Ricci, W. Wu, and B. Zhou, "UrbanVerse: Scaling urban simulation by watching city-tour videos," in *Proc. Int. Conf. Learning Representations (ICLR)*, 2026.

[6] Z. Wang, S. Lee, Y. Xu, and G. H. Lee, "Image2Sim: Scaling embodied navigation via generative neural simulator," *arXiv preprint arXiv:2607.05765*, 2026.

[7] Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2017.

[8] M. Everett, Y. F. Chen, and J. P. How, "Motion planning among dynamic, decision-making agents with deep reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2018.

[9] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2019.

[10] Z. Xie and P. Dames, "DRL-VO: Learning to navigate through crowded dynamic scenes using velocity obstacles," *IEEE Transactions on Robotics*, vol. 39, no. 4, pp. 2700–2719, 2023.

[11] S. Liu, P. Chang, Z. Huang, N. Chakraborty, K. Hong, W. Liang, D. L. McPherson, J. Geng, and K. Driggs-Campbell, "Intention aware robot crowd navigation with attention-based interaction graph," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2023, pp. 12 015–12 021.

[12] S. Liu, H. Xia, F. C. Pouria, K. Hong, N. Chakraborty, Z. Hu, J. Biswas, and K. Driggs-Campbell, "HEIGHT: Heterogeneous interaction graph transformer for robot navigation in crowded and constrained environments," *IEEE Transactions on Automation Science and Engineering*, vol. 23, pp. 1211–1230, 2026.

[13] N. Kumar, A. Singha, L. Nanwani, D. Potdar, T. R, F. Rastgar, S. Idoko, A. K. Singh, and K. M. Krishna, "CrowdSurfer: Sampling optimization augmented with vector-quantized variational autoencoder for dense crowd navigation," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2025, pp. 16 854–16 860.

[14] H. Karnan, A. Nair, X. Xiao, G. Warnell, S. Pirk, A. Toshev, J. Hart, J. Biswas, and P. Stone, "Socially CompliAnt navigation dataset (SCAND): A large-scale dataset of demonstrations for social navigation," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 807–11 814, 2022.

[15] D. M. Nguyen, M. Nazeri, A. Payandeh, A. Datar, and X. Xiao, "Toward human-like social robot navigation: A large-scale, multi-modal, social human navigation dataset," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2023.

[16] N. Hirose, D. Shah, A. Sridhar, and S. Levine, "SACSoN: Scalable autonomous control for social navigation," *IEEE Robotics and Automation Letters*, vol. 9, no. 1, pp. 49–56, 2024.

[17] N. Pérez-Higueras, R. Otero, F. Caballero, and L. Merino, "HuNavSim: A ROS 2 human navigation simulator for benchmarking human-aware robot navigation," *IEEE Robotics and Automation Letters*, vol. 8, no. 11, pp. 7130–7137, 2023.

[18] V. Shcherbyna, L. Kästner, D. A. Do, J. Wang, H. G. Nguyen, T. Seeger, A. Martban, Z. Shen, H. Zeng, N. Trinh, and E. Wiese, "Arena-Bench 2.0: A comprehensive benchmark of social navigation approaches in collaborative environments," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2025, pp. 9202–9209.

[19] A. Biswas, A. Wang, G. Silvera, A. Steinfeld, and H. Admoni, "SocNavBench: A grounded simulation testing framework for evaluating social navigation," *ACM Transactions on Human-Robot Interaction*, vol. 11, no. 3, pp. 1–24, 2022.

[20] G. Chhablani, X. Ye, M. Z. Irshad, and Z. Kira, "EmbodiedSplat: Personalized real-to-sim-to-real navigation with Gaussian splats from a mobile device," in *Proc. IEEE/CVF Int. Conf. Computer Vision (ICCV)*, 2025.

[21] N. Hirose, C. Glossop, A. Sridhar, O. Mees, and S. Levine, "Lelan: Learning a language-conditioned navigation policy from in-the-wild video," in *Proceedings of The 8th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, P. Agrawal, O. Kroemer, and W. Burgard, Eds., vol. 270. PMLR, 06–09 Nov 2025, pp. 666–688. [Online]. Available: https://proceedings.mlr.press/v270/hirose25b.html

[22] X. Liu, J. Li, Y. Jiang, N. Sujay, Z. Yang, J. Zhang, J. Abanes, J. Zhang, and C. Feng, "CityWalker: Learning embodied urban navigation from web-scale videos," in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR)*, 2025.

[23] Y. Mei, Y. Yang, L. Guo, Q. Wang, M.-M. Yu, X. He, W. Wu, and J. Liu, "UrbanNav: Learning language-guided embodied urban navigation from web-scale human trajectories," in *Proc. AAAI Conf. Artificial Intelligence (AAAI)*, vol. 40, no. 22, 2026, pp. 18 505–18 513.

[24] G. Seneviratne, Y. Abeysinghe, J. An, V. Shende, R. Kumar, and D. Manocha, "VEGA: Learning navigation VLAs from in-the-wild egocentric video with geometric trajectory supervision," *arXiv preprint arXiv:2606.18426*, 2026.

[25] H. Lin, S. Chen, J. H. Liew, D. Y. Chen, Z. Li, Y. Zhao, S. Peng, H. Guo, X. Zhou, G. Shi, J. Feng, and B. Kang, "Depth anything 3: Recovering the visual space from any views," in *International Conference on Learning Representations*, C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust, Eds., vol. 2026, 2026, pp. 141 261–141 285. [Online]. Available: https://proceedings.iclr.cc/paper_files/paper/2026/file/e4cd50120b6d7e8daff1749d6bbaa889-Paper-Conference.pdf

[26] J. Wang, D. Liu, J. Chen, J. Da, N. Qian, M. M. Tram, and H. Soh, "GeNIE: A generalizable navigation system for in-the-wild environments," *IEEE Robotics and Automation Letters*, 2025.

[27] G. Jocher and J. Qiu, "Ultralytics YOLO11," https://github.com/ultralytics/ultralytics, 2024.

[28] N. Aharon, R. Orfaig, and B.-Z. Bobrovsky, "BoT-SORT: Robust associations multi-pedestrian tracking," *arXiv preprint arXiv:2206.14651*, 2022.

[29] J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu, "RoFormer: Enhanced transformer with rotary position embedding," *Neurocomputing*, vol. 568, 2024.

[30] P.-W. Chou, D. Maturana, and S. Scherer, "Improving stochastic policy gradients in continuous control with deep reinforcement learning using the beta distribution," in *Proc. Int. Conf. Machine Learning (ICML)*, 2017.

[31] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[32] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, and A. R. Zamir, "On evaluation of embodied navigation agents," *arXiv preprint arXiv:1807.06757*, 2018.

[33] J. Wang and H. Soh, "Probable object location (POLO) score estimation for efficient object goal navigation," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2024, pp. 5221–5227.

[34] S. Macenski, F. Martín, R. White, and J. G. Clavero, "The marathon 2: A navigation system," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2020.